

---

# Program Verification

An Application of Predicate Logic

# Program verification

---

- Introduction and Hoare triples
- Inference rules for verification
- Loop invariants

# The Idea

---

- Verification:
  - (Does the program run and terminate?)
  - Does it accomplish its goal? (partially correct)
- Use logic to:
  - Specify the initial state of a program
  - Express how statements change states
  - Specify the final state = goal of the program
  - Prove that we can...start at the start, go to the goal, stop

# Generic Statements

---

Simple statement types to show the idea:

- **Assignment**
- **Sequencing**
- **If-then-else**
- **While- do**

# Hoare Triple

---

- A proposition about a program statement
- Specifies the changes in what is true
- General Form
  - $p \{S\} q$
  - $p$  and  $q$  are propositions
  - $S$  is a program statement

$$p \{S\} q$$

---

- If  $p$  is true
- and statement  $S$  is executed
- then  $q$  is true after execution, assuming  $S$  terminates
- $p$  is the precondition and  $q$  is the post condition.
- $S$  could be a single statement or an entire program

# Inference Rule for Verification

## Assignment

---

$$p(e) \{v \leftarrow e\} p(v)$$

- $\leftarrow$  is the assignment operator
- If  $e$  satisfies predicate  $p$  before execution, then  $v$  satisfies  $p$  afterward.
- Examples:
  - **Odd(3) {x  $\leftarrow$  3} Odd(x)**
  - **Odd(y) {x  $\leftarrow$  y+2} Odd(x)**
  - **Odd(x) {x  $\leftarrow$  x+1} Even(x)**
    - $X_{\text{after}} = X_{\text{before}} + 1$

# Inference Rules for Verification

## Sequence

---

$$p \{S1\} q$$
$$\underline{q \{S2\} r}$$
$$p \{S1, S2\} r$$

- $q$ : postcondition for  $S1$  and precondition for  $S2$
- Example

$$(x = 1) \quad \{y \leq 3\} \quad (x=1) \wedge (y=3)$$
$$\underline{(x=1) \wedge (y=3) \quad \{z \leq x + y\} \quad (z=4)}$$
$$(x=1) \quad \{y \leq 3; z \leq x + y\} \quad (z = 4)$$

# Another Example

---

$$\begin{array}{ccc} (x=y) & \{x \sqcap x+1\} & (x = y+1) \\ \hline (x = y+1) & \{y \sqcap y+1\} & (x = y) \\ \hline (x=y) & \{x \sqcap x+1; y \sqcap y+1\} & (x = y) \end{array}$$

- $x = y$  “Invariant”
  - $x=y$  holds on entry and it holds on exit
  - $x \neq y$  is possible during execution
  - Subscripts avoid confusion

# Subscripts

---

$$x_0 = y_0$$

$$x_1 = x_0 + 1 \quad \text{and} \quad y_1 = y_0 \quad \text{step 1 - statement 1}$$

$$y_2 = y_1 + 1 \quad \text{and} \quad x_2 = x_1 \quad \text{step 2 - statement 2}$$

$$x_2 = x_1 = x_0 + 1 = y_0 + 1 = y_1 + 1 = y_2$$

$$x_2 = y_2$$

# Inference Rules for Verification

## Conditionals

---

### If-then

$$(p \sqcap B) \{S\} (q)$$

$$\underline{(p \sqcap \neg B) \sqcap (q)}$$

$$p \{\text{if } B \text{ then } S\} q$$

- If p is true then either
  - If B is True q **becomes** true when S executes or
  - If B is False q **is already** True (provable)

### If-then-else

$$(p \sqcap B) \{S1\} (q)$$

$$\underline{(p \sqcap \neg B) \{S2\} (q)}$$

$$p \{\text{if } B \text{ then } S1 \text{ else } S2\} q$$

# Example

---

$$(p \sqcap B) \{S\} (q)$$

$$\underline{(p \sqcap \neg B) \sqcap (q)}$$

$$p \{ \text{if } B \text{ then } S \} q$$

- Statement:  $\{ \text{if } y < x \text{ then } y \sqcap x \}$
- Precondition  $x = 7$
- Show the postcondition  $y \geq 7$
- $B \equiv y < x$  so  $\neg B \equiv y \geq x$
- $S \equiv y \sqcap x$

$$(x = 7 \sqcap y < x) \{y \sqcap x\} (y \geq 7)$$

$$\underline{(x = 7 \sqcap y \geq x) \sqcap (y \geq 7)}$$

$$x = 7 \{ \text{if } y < x \text{ then } y \sqcap x \} (y \geq 7)$$

# Verify Code for $|X|$

---

$(p \sqcap B) \{S1\} (q)$

$(p \sqcap \neg B) \{S2\} (q)$

$p \{ \text{if } B \text{ then } S1 \text{ else } S2 \} q$

$(\text{TRUE} \sqcap x < 0) \quad \{ \text{abs} \sqcap -x \} \quad (\text{abs} = |x|)$

$(\text{TRUE} \sqcap x \geq 0) \quad \{ \text{abs} \sqcap x \} \quad (\text{abs} = |x|)$

$(\text{TRUE}) \{ \text{if } x < 0 \text{ then } \text{abs} \sqcap -x \text{ else } \text{abs} \sqcap x \} (\text{abs} = |x|)$

# Loop Invariants

## Example

---

**While:**

$$\frac{(p \wedge B) \quad \{S\} \quad p}{p \quad \{\text{while } B \text{ do } S\} \quad (p \wedge \neg B)}$$

- **The oddball puzzle:**
- A bag has M white and N black balls.
- You repeatedly randomly remove two.
- If opposite color, put the white one back.
- If same color, put a black one back (you must have extras).
- Eventually one ball is left since we remove one ball on each turn
- What color is it?

# Selections

---

- 17W 10B: draw 1W & 1B then return W
- 17W 9B: draw 2B then return B
- 17W 8B: draw 2W then return B
- 15W 9B
- .....

# Solution Using Loop Invariant

---

- $S$  = the entire code of the selection, replacement counter update loop.

while ( $m+n > 1$ ) do

    choose a ball - if white decrement white count  
                  else decrement black count.

    choose a ball - if white decrement white count  
                  else decrement black count

    if the choices match increment black count  
        else increment the white count

# Applying the Inference Rulee

---

- $p$  = the loop invariant is  $(\text{odd}(\text{white count}))$ 
  - which must equal  $\text{odd}(\text{WHITECOUNT})$ .
  - Exhaust the possible cases
    - If both are black, we return a black and the parity of white is unchanged.
    - If both are white, we return a black and the parity of white remains the same as it was before choosing (although the count is reduced by two).
    - If we choose one of each, we return the white and the parity remains unchanged.
- $B$  = condition for staying in loop,  $m+n > 1$ .
- $p \{ \text{while } B \text{ do } S \} (p \square \square B)$
- One ball is left and the parity of the white count is the same as it was when we started
  - Since we started with an odd number of white balls, we must have a white ball left to guarantee the parity is preserved

# Another Example for n!

---

$(p \sqcap B) \{S\} p$   
 $p \{ \text{while } B \text{ do } S \} (p \sqcap \sqcap B)$

- Computing n!

i  $\sqcap$  1;

f  $\sqcap$  1;

while i < n do

begin

i  $\sqcap$  i+1;

f  $\sqcap$  f\*i;

end

# n!

- 
- Compute n! in the variable f.
  - The loop invariant includes  $f = i!$  at the beginning of the loop,
  - Also, make  $i=n$  at loop exit, so that  $f = n!$ .
  - Entire loop invariant  $p : (f = i!) \wedge (i \leq n)$ .
  - Must have  $n \geq 1$  initially, to make  $p$  true.
  - $p$  is a loop invariant, with loop condition,  $B = (i < n)$  if:

$(f = i!) \wedge (i \leq n) \wedge (i < n)$

$\{i \leq i+1; f \leq f*i\}$

$(f = i!) \wedge (i \leq n) \wedge (i < n)$

# What the Rule Gives Now

---

$(f = i!) \wedge (i \leq n)$

{while( $i < n$ ) do S}

$(f = i!) \wedge (i \leq n) \wedge (i \geq n)$

... so when the loop terminates:  $i = n$ , and so  $f = n!$ .

# Summary

---

- Introduction and Hoare triples
- Inference rules for verification
  - Assignment
  - Sequencing
  - If-then-else
  - While- do
- Loop invariants